MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A
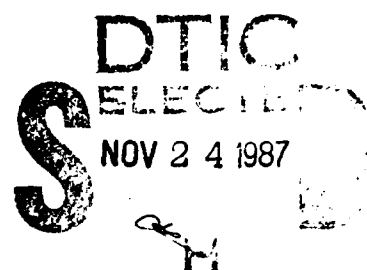
ⓐ

NAVAL POSTGRADUATE SCHOOL
Monterey, California

DTIC
ELECTE
NOV 2 4 1987

THESIS

DEPLOYMENT PLANNING: A LINEAR PROGRAMMING
MODEL WITH VARIABLE REDUCTION

by

K. Steven Collier

September 1987

Thesis Advisor: Richard E. Rosenthal

87 11 14 036

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION	1b RESTRICTIVE MARKINGS
Unclassified	

2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT
	Approved for public release;
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	Distribution is unlimited

4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)

6a NAME OF PERFORMING ORGANIZATION	6b OFFICE SYMBOL (If applicable)	7a NAME OF MONITORING ORGANIZATION
Naval Postgraduate School	55	Naval Postgraduate School

6c ADDRESS (City, State, and ZIP Code)	7b ADDRESS (City, State, and ZIP Code)
Monterey, California 93943-5000	Monterey, California 93943-5000

8a NAME OF FUNDING/SPONSORING ORGANIZATION	8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c ADDRESS (City, State and ZIP Code)	10 SOURCE OF FUNDING NUMBERS			
	PROGRAM ELEMENT NO	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification)

DEPLOYMENT PLANNING: A LINEAR PROGRAMMING MODEL WITH VARIABLE REDUCTION

12 PERSONAL AUTHOR(S)

COLLIER, K. Steven

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Master's thesis	FROM ___ TO ___	1987 September	72

16 SUPPLEMENTARY NOTATION

17	COSATI CODES		18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Joint Deployments, Linear Programming, Variable Reduction, GAMS, Multicommodity Capacitiated Transhipment Problem.

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

The United States Armed Forces must be capable of deploying to areas of operations anywhere in the world. Planning for these deployments is the responsibility of the Joint Deployment Agency, MacDill Air Force Base, Tampa, Florida. Deployment plans are large and complex. A straightforward linear programming model of a deployment plan can easily exceed 700 million decision variables.

This study outlines the development of a system used to assist planners in determining deployment plan feasibility and in selecting modes of transportation. The system consists of a data input array, an algorithm to eliminate all unusable variables, and a linear programming model.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT	21 ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	Unclassified

22a NAME OF RESPONSIBLE INDIVIDUAL	22b TELEPHONE (Include Area Code)	22c OFFICE SYMBOL
Prof. Richard Rosenthal	408-646-2795	55R1

DD FORM 1473, 84 MAR   83 APR edition may be used until exhausted   SECURITY CLASSIFICATION OF THIS PAGE

All other editions are obsolete

BLOCK 19. ABSTRACT (continued)

The largest scenario considered in this study is a 90-day deployment plan with 90 movement requirements, 9 types of lift assets, traveling between 22 ports. This corresponds to a linear programming model with 35 million decision variables. The variable reduction algorithm reduced the number of decision variables to 11,100, and an optimal solution was found in a total computation time (input, reduction, optimization, output) time of 6.5 minutes.

**Deployment Planning: A Linear Programming Model
With Variable Reduction**

by

K. Steven Collier
Captain, United States Army
B.S., United States Military Academy, 1977

Submitted in partial fulfillment of the
requirements for the degree of

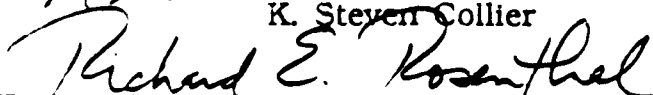MASTER OF SCIENCE IN
OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 1987

Author: _____
K. Steven Collier
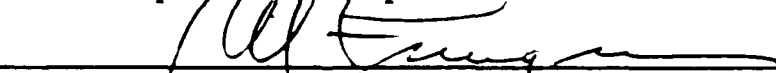
Approved by: _____
Richard E. Rosenthal, Thesis Advisor

_____
Siriphong Lawphongpanich, Second Reader

_____
Peter Purdue, Chairman,
Department of Operations Research

_____
James M. Fremgen,
Acting Dean of Information and Policy Sciences

3

ABSTRACT

The United States Armed Forces must be capable of deploying to areas of operations anywhere in the world. Planning for these deployments is the responsibility of the Joint Deployment Agency. MacDill Air Force Base. Tampa. Florida. Deployment plans are large and complex. A straightforward linear programming model of a deployment plan could easily exceed 700 million decision variables.

This study outlines the development of a system used to assist planners in determining deployment plan feasibility and in selecting modes of transportation. The system consists of a data input array. an algorithm to eliminate all unusable variables. and a linear programming model.

The largest scenario considered in this study is a 90-day deployment plan with 90 movement requirements. 9 types of lift assets. traveling between 22 ports. This corresponds to a linear programming model with 35 million decision variables. The variable reduction algorithm reduced the number of decision variables to 11.100. and an optimal solution was found in a total computation time (input. reduction. optimization. output) time of 6.5 minutes.

4

TABLE OF CONTENTS

5

ACKNOWLEDGMENTS

7

I. INTRODUCTION

The complexity and magnitude of deploying US forces to an overseas area requires careful and thorough coordination. Sound deployment planning is critical to the successful execution of any deployment. This thesis develops a linear programming optimization model which will assist deployment planners in the evaluation and development of more efficient deployment plans. The model developed in this study is an alternative approach to the model currently being developed by the Joint Deployment Agency (JDA).

The JDA model is the System for Closure Optimization Planning and Evaluation (SCOPE). This model has been in the developmental stage for five years. The primary developer of the SCOPE model has been a team led by Professors John J. Jarvis and H. Donald Ratliff of the Georgia Institute of Technology (GT). All future references to their model will be as SCOPE-GT. The linear programming (LP) model developed in this study will be referred to as SCOPE-NPS.

The SCOPE-GT model being used at the JDA is not a "stand-alone" model. It is a component of the Mode Optimization and Deployment Estimation Subsystem (MODES). Furthermore, MODES is a subsystem of the Joint Deployment System (JDS). The primary developer of the MODES subsystem is the Computer Sciences Corporation (CSC). [Ref. 1]

8

Some of the problems being experienced at the JDA with the MODES subsystem are outlined in the next chapter. Suffice it to say that there are problems and that, due to the complexity of combined JDS, MODES, and SCOPE–GT systems, these problems have been hard to identify. A potential problem area has been identified as the performance of a Benders decomposition algorithm in the SCOPE–GT model. Problems of solution acuracy and computation time associated with this formulation provided the primary impetus to develop alternatives.

The efforts to develop new approaches were undertaken in two simultaneous studies in the Master of Science in Operations Research program at the Naval Postgraduate School (NPS). The development of SCOPE–NPS is presented in this thesis. The second study was conducted by Captain Michael Lally and is presented in his thesis [Ref. 2]. The purpose of this second study was to develop an integer programming formulation that would correct deficiencies in the way SCOPE–GT represents sea transport. Initially, the goal was to have these two efforts merge into a single model. Although each study has resulted in an operating model, the goal of combining them has yet to be accomplished.

The results of this study demonstrate that small– and medium-sized deployment problems can be realistically modeled, and solved utilizing a linear programming formulation coupled with a "variable reduction" algorithm. The largest model tested in this study considered a medium–sized deployment problem with 35 million decision

9

variables. The key to SCOPE–NPS's ability to solve a problem of this size is a preprocessing algorithm which "intelligently" reduces the number of decision variables without affecting the optimal solution.

This thesis details how SCOPE–NPS was developed into a system consisting of three components: the Data Input Array (DIA), the Arc Reduction Algorithm (ARA), and the Matrix Generator, as depicted in Figure 1-1.

Chapter 2 of this thesis provides the following background information: a description of a deployment plan, the responsibilities of deployment planning agencies, a description of the deployment planning environment, and a brief introduction to the SCOPE–GT model. The SCOPE–GT introduction includes a discussion concerning the decomposition formulation and some of the problems currently being experienced.

The SCOPE–NPS model is defined and formulated in Chapter 3. Chapter 4 presents the algorithm for reducing the number of decision variables in a deployment problem. Chapter 5 presents the results of SCOPE–NPS model tests and suggests several model enhancements which may be incorporated into some future studies.

```
┌─────────────┐
│    Data     │        ⎧ Formatted Data Array
│ Input Array │        ⎪ Provides:  Deployment Problem Size
│   (DIA)     │        ⎨ Movement Requirment Data
└─────────────┘        ⎪ Lift Asset Data
                       ⎪ Port Data
                       ⎩ (See Chapter V)

┌─────────────┐        ⎧ FORTRAN 77 Program
│    Arc      │        ⎪ (See Chapter IV)
│  Reduction  │        ⎨ Reduces the complete set of posible
│  Algorithm  │        ⎪ decision-variables down to a much
│   (ARA)     │        ⎩ smaller set of necessary variables.
└─────────────┘

┌─────────────┐        ⎧ Three FORTRAN 77 Subroutines
│   Matrix    │        ⎪ called by ARA.  The MG converts the
│  Generator  │        ⎨ decision variables provided by the
│    (MG)     │        ⎪ ARA via the Mathematical Formula-
└─────────────┘        ⎪ tion (see Chapter III) to the MPS
                       ⎩ Format (see Chapter V).

          IF
    All Movement
No  Requirements
    Have Been
    Accounted
       For

        Yes

┌─────────────┐
│   MPS III   │
│ LP Optimizer│
└─────────────┘
```
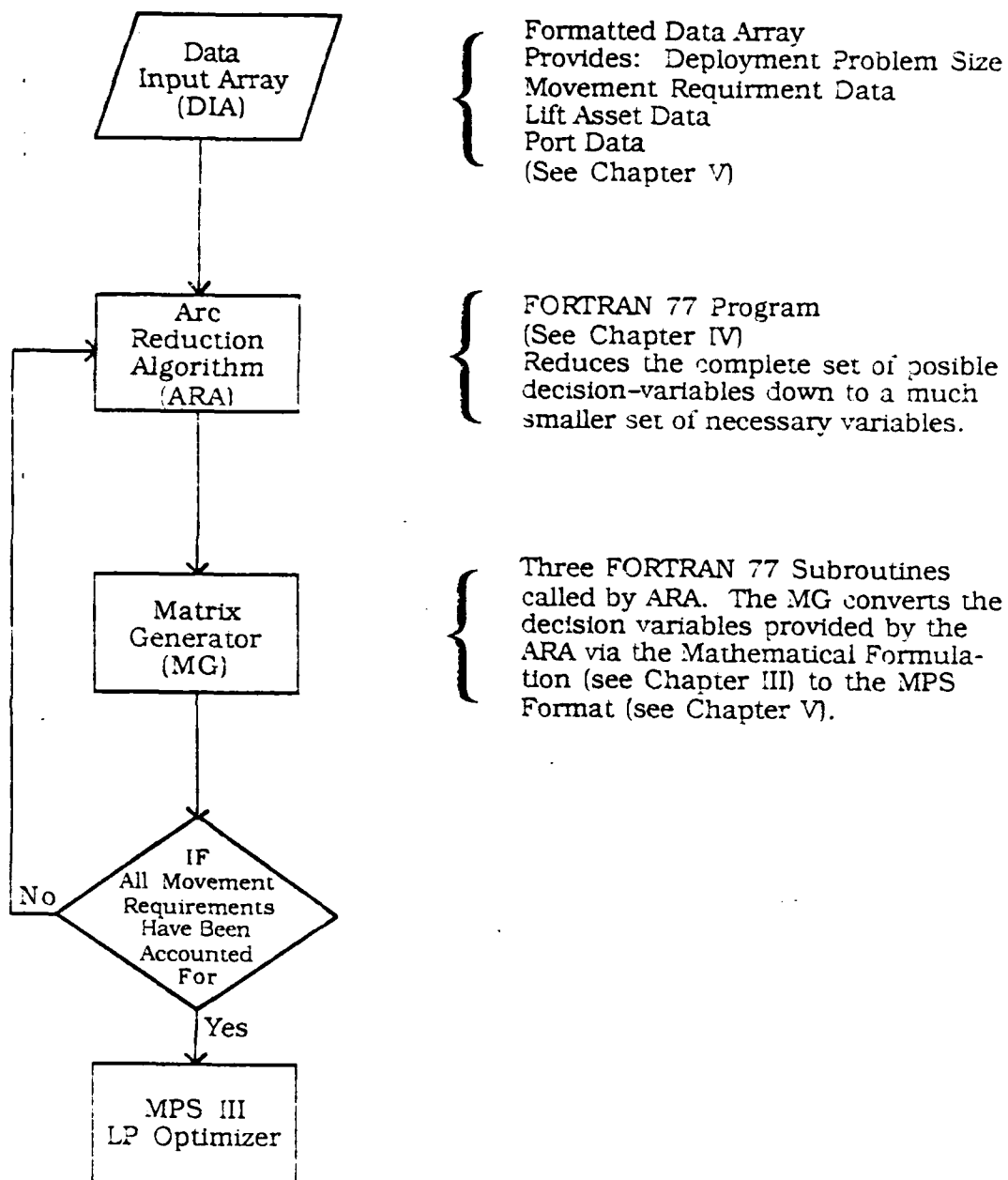
Figure 1.1

SCOPE-NPS

11

II. BACKGROUND

A. DEPLOYMENT PLAN DESCRIPTION

During peacetime exercises or periods of conflict. US forces (Army. Navy. Air Force. and Marines) must be capable of moving from their home bases to areas of operations anywhere in the world. The movement of these forces is called a deployment. A deployment may involve moving 20 soldiers from Ft. Bragg. North Carolina. for a week of training in Panama. or it may involve moving 100,000 soldiers from several US bases for the defense of Europe. Plans for these deployments may evolve over a period of years or may be conceived and executed in a matter of hours.

The planning, coordination. and execution of any deployment plan may be one of the most difficult of all military operations. In the worst case. a unit and its equipment may be deployed to a location occupied by enemy forces. Initially. we will not have access to either airports or seaports. As these facilities become available. reinforcements and resupply operations must commence immediately. The exact timing and order in which units, equipment, and resupplies arrive is a key element in insuring the success of any deployment. The development of a deployment plan is the responsibility of the commander who must execute the deployment. The commander's plan for the deployment is called an "Operations Plan" (OPLAN). His schedule. once refined. is called the "Time-Phased Force Deployment List" (TPFDL).

B. INPUTS TO SCOPE–NPS PROBLEM

For the purpose of this study, the key elements of the OPLAN and TPFDL have been capsulized into the following essential input items:

1. Movement Requirements (MR)

A movement requirement constitutes an "order" for some commodity to be transported and delivered according to a specified schedule. Each MR's specifications include:

- MR description (passengers, bulk cargo, fuel, ammunition, etc.)

- Date the MR is available to ship

- Date the MR is required to be delivered

- MR priority

- MR size and/or quantity

- MR origin

- MR destination

2. Available Ports

Ports may be airports, seaports, or rail or truck terminals. Ports may also be classified as Ports of Embarkation (POE) or Ports of Debarkation (POD). Port data includes:

- Port cargo-handling capacity for both loading and unloading. This capacity is usually expressed in short tons (stons) per time period.

- Port access restrictions. Seaports may only allow a certain number of ships to be in port at one time, and these ships cannot exceed a certain size. Airports have similar restrictions.

13

3. Lift Assets

Lift assets include: cargo and passenger planes, various types of ships, trains, trucks, etc. Lift data includes:

- Quantity of each asset type available during each time period.

- Capacity of each asset type (stons).

- Cycle time for each asset type between two ports. This time includes loading, unloading, refueling, and scheduled maintenance time.

Given this list of data, the SCOPE-NPS system attempts to meet the required delivery schedule while simultaneously optimizing the use of all lift assets.

The number of possible decision variables associated with an optimization model of a deployment problem can be tremendous. A realistic problem size is 500 movement requirements, 10 lift assets (C141, C5, RORO, Breakbulk, etc), 40 POEs, 40 PODs, and 90 days. If the model considered all possible combinations of movement request, asset type, POE, POD, and time period, there would be 720 million variables. It is clearly imperative for any modeling system to significantly reduce the number of decision variables explicitly considered.

C. RESPONSIBILITIES OF THE JOINT DEPLOYMENT AGENCY

The need for a more coordinated effort between our separate branches of service has been evidenced in every joint US forces operation since World War II. A typical example occurred during a recent exercise when the support operation was forced to a complete standstill. In this case, too many planes had landed at a small airport.

The result was a logjam of airplanes that precluded any planes from landing or taking off.

Another recent example occurred during the British invasion of the Falkland Islands. While the invasion force was en route to the Falklands. a detailed review of the rapidly prepared deployment plan revealed a major deficiency. Although the key supply ship for the invasion had been loaded with the requisite supplies. he ship had been loaded in reverse order. This discovery resulted in an unscheduled delay which required the invasion force to offload and properly reload the supply ship.

Recognizing that our ability to conduct well-coordinated joint deployments would be critical in any major operation. the Joint Deployment Agency (JDA) was established in March 1979. The JDA was to be the single point of contact for deployment planning and coordination.

The Joint Deployment Agency's mission is to support the Joint Chiefs of Staff (JCS) and Commanders in Chief (CINCs) in planning for and executing deployments. The JDA is responsible for coordinating the actions of deploying units and common-user land. air. and sealift movements. The Military Traffic Management Command (MTMC) is responsible for movement within the continental United States. he Military Airlift Command (MAC) for aerial movements. and the Military Sealift Command (MSC) for movement by sea. The JDA also serves as the focal point for information associated with deployment decisions.

D. DEPLOYMENT PLANNING ENVIRONMENT

In analyzing the SCOPE model, it is important to recognize the level of decision making for which it is intended. Its purpose is to provide the JDA with the ability to "assess potential deployment feasibility problems" and to assist in transportation allocation decision making [Ref. 3]. This level of decision making is referred to as "closure planning," and must be distinguished from decisions concerning how each asset is "scheduled" [Ref. 4]. Deployment planning is usually conducted in a deliberate mode. In the deliberate mode, deployment OPLANs are reviewed, refined, and updated whenever conditions change. Deployment planning may also occur in a crisis environment. During a crisis, decisions must be made and plans selected or written in a matter of hours. To accommodate the worst-case (crisis) scenario, any model developed for the purpose of analyzing a deployment plan should be required to support the decision makers within a four-hour time window [Ref. 3:pp. 1-2].

E. CURRENT MODEL (SCOPE-GT)

1. Purpose and Development

The primary research attempting to solve this large deployment problem has come from a team led by Professors John J. Jarvis and H. Donald Ratliff of the Georgia Institute of Technology. During the past five years, their effort has been to:

- Examine deployment planning in a crisis action environment from a modeling perspective;

16

- Assess available methodology and modeling concepts for application to the crisis action environment;

- Develop concepts and methodology for closure optimization; and

- Develop a system design within which these models would function.

Jarvis and Ratliff describe a hierarchy of four levels in which the models would function. Decisions and assumptions made at the higher levels guide and constrain decisions at the lower levels. Violations of these constraints cannot occur unless the higher level modifies or changes the constraining decision or assumptions. The lower the level, the greater the detail involved in the planning process.

The highest level is the closure planning level. The primary purpose of this level is to aid the decision maker in developing a general movement plan which will satisfy the military objectives and can be supported by the available transportation system. A general movement plan includes mode. POE. POD. assignment of movement requirements, timing of movements, degree of flexibility allowed at lower levels, and the manner in which movement requirements can be split for transportation. The decisions made at this level are the most important because they guide and constrain all future decisions.

The second level is the system loading/coordination level. Its purpose is to insure efficient utilization of the transportation system in carrying out the general movement plan developed in level one. At this level, they search for and attempt to resolve problem areas and develop more detail regarding movements. Additionally, it provides information and coordination necessary for transition from the top

17

level to the detailed scheduling by transportation operating agencies in level three.

The third level is where detailed schedules are constructed by MTMC, MAC, and MSC. These transportation operating agencies are given specific movement requirements, suggested lift assets, POE, POD, and the required delivery dates.

The level four system is for monitoring the development and implementation of the deployment plan. This four-level system is a dynamic planning system that provides for feedback, updates, and modifications as the plan proceeds. [Ref. 4:pp. 5-17]

2. SCOPE-GT Model Description

The main thrust of the Georgia Tech research has been on level one, where the general movement plan is developed. They decided the best way to solve the deployment problem was to use decomposition. They broke the problem into two subproblems— a channel configuration and a movement requirement assignment problem. The problems are connected through a set of linking constraints. The decomposition method first generates the solution to the channel configuration model. With the linking constraints fixed, the movement requirement assignment problem is solved. The results of this model generate a linking constraint that is passed back to the channel configuration model, which is solved again. This process is repeated until the solutions converge to optimality or it can be stopped at the user's discretion if time is limited.

18

The system the Georgia Tech team is developing to implement this approach consists of three major components:

a. The preprocessor. which loads the applicable operations plan into the data base. loads the movement requirements that support the operations plan. coordinates information. and generates the necessary parameters such as port capacities. lift capacities. and transit times. Additionally. the operations plan can be modified or a brand new plan can be constructed.

b. The solver and SCOPE–GT model. which is discussed in the next section.

c. The postprocessor. which generates the output that can be displayed with tabular data and graphics.

In the search for appropriate solvers. Georgia Tech looked for solution methodologies which would be most suitable for large deployment problems. The appropriate solver would. as a minimum. consider the following:

- Structure and sparsity of the deployment network
- Computational speed
- Storage requirements

The movement requirement assignment problem has a pure network structure therefore it can be best solved using a network solver. For the channel configuration model. the Georgia Tech team chose a solver for networks with side constraints. The number of side constraints is sometimes more than this solver can effectively handle. so the Georgia Tech team may switch to a linear program solver in the

future. The two problems are linked together with Bender's decomposition method. [Ref. 4:pp. 44-54]

3. Current Model Deficiencies

The current model is experiencing several problems. It takes a long time to converge and at times will not converge to the optimal answer. While on experience tour at JDA in November 1986, a small test problem was submitted to the current model and it produced an obviously suboptimal answer. In this small problem, every movement requirement was available to be shipped during the first time period. All transportation assets were also available during the first time period and could easily cycle between the POEs and PODs in one time period. However, the required delivery dates for each movement request were during the first time period. An obvious optimal solution would have been to begin deliveries during the first time period. However, the SCOPE-GT solution did not make its first delivery until the third time period. Research is continuing in an attempt to discover the source of the convergence problem.

The current model takes over eight hours to solve medium-size deployment problems. This is not fast enough for crisis planning. Current research is investigating a "hot restart" capability, aggregation of movement requirements, suboptimal stopping rules, a method to generate arcs is needed, and arc reduction methods.

A third area of concern is the method of modeling sealift. The model assumes a continuous flow rate. The associated channel

20

concept can best be understood by likening the channel and its capacity to a pipe with water passing through it at a given flow rate.

The Georgia Tech research team makes a good argument for the channel concept and continuous flow rate when applied to airlift. The airlift cycle times are relatively small when compared to the time horizon and the delivery effect is "smoothed" over time. However, they try to apply the same argument to sealift. The following example shows how a continuous flow rate makes sealift appear unrealistic. Consider a ship with a capacity of 10,000 stons and a ten-day cycle time between two ports. The continuous flow solution would allow this ship to make ten consecutive 1,000-ston deliveries instead of one 10,000-ston delivery. The users of the model do not want cargo "flowing" through seaports. They prefer discrete shipments. Discrete shipments more realistically portray ship departures and arrivals. [Ref. 5]

After a six-week evaluation of the SCOPE-GT model, Captain Lally and I decided to take a new look at the problem and determine alternate methods that could be used to solve the deployment problem.

As stated earlier, Captain Lally chose to develop a model that can be used to allocate strategic sealift resources. His research shows that integer programming with variable reduction methods is a viable approach to solving the sealift allocation problem. This study focuses on a linear programming model designed to: (1) determine OPLAN feasibility, and (2) optimally allocate air and sea lift assets.

III. MODEL DEVELOPMENT AND DESCRIPTION

A. MODEL DESCRIPTION

The deployment model is a multicommodity capacitated trans-shipment problem (MCTP). These problems occur in many forms and fall into the class of minimum cost network flow problems [Ref. 6]. Assad [Ref. 7] and Kennington [Ref. 8] discuss the MCTP and the various methods which have been developed to solve them. A description of the minimum cost flow problem along with the node-arc formulation is given by Bradley, Brown, and Graves [Ref. 9] and Bazaraa and Jarvis [Ref. 10]. In most cases, the purpose of these models is to minimize shipment cost. In the current context of deployment scenarios, minimizing shipping cost, or efficiently utilizing assets, must be balanced against the strict adherence to a time schedule. If this time schedule cannot be met, the solver should identify which movement requirements can and cannot be met. It should also provide information as to where additional resources (ports, planes, ships, etc.) can be most efficiently allocated to make the problem feasible.

In a deployment problem, timing is critical. This requires representing each individual movement request as a single commodity. All commodities must share the same set of assets, so they are bound together by the presence of joint capacity constraints. These joint capacities preclude the use of pure network solvers. However, MCTP still possesses a block diagonal structure which lends itself to

22

decomposition. Bazaraa and Jarvis [Ref. 10:pp. 492-494] provide a description of the coefficient matrix and its block diagonal form and discuss how it lends itself to decomposition.

The major approaches to decomposing these large problems were formalized in the Dantzig-Wolfe decomposition principle [Ref. 10:p. 351] and in Benders decomposition method [Ref. 11; Ref. 4:pp. 44-54]. As described earlier. SCOPE-GT utilizes a formulation based on Benders decomposition.

A guideline of this research was to restrict the approach to linear programming formulations which could. in reasonable time. provide feasible. usable solutions without decomposition or other advanced algorithms. While a direct LP approach may not handle the largest of deployment problems. such as multitheater planning, we believe it has the potential to solve the great percentage of plans which fall into the small or medium size categories. Moreover. if the viability of this approach is demonstrated. then the development effort required for operational implementation is substantially less costly and risky than the decomposition approach.

The linear programming model presented in this thesis incorporates the following key attributes of the deployment problem:

- Provide gross feasibility estimates
- Minimize deviations from required delivery dates.
- Minimize shipping cost (minimize shiping time on cheapest available asset).
- Select mode of transportation.

23

- Represent sealift more realistically.

- Provide for prioritized delivery of movement requests.

- Observe port capacities.

- Observe lift asset capacities.

- Provide for an elastic/feasible solution (see the next section for an explanation of this attribute).

- Solve realistically sized problems.

B. MATHEMATICAL FORMULATION

The basic model is presented here in a node-arc formulation. The classical formulation has been augmented with the requisite lift asset and port capacity constraints.

Indices:

r = Movement requirement (commodity).

a = Lift asset type.

i,j = Ports of embarkation and debarkation (source, destination, or transshipment nodes).

t = Time period.

Data (grouped by category):

Movement Requirement Data:

ALD(r)   = Time period movement requirement r is available to load.

RDD(r)   = Time period movement requirement r is required to be delivered

MD(r,t)   = $RDD(r) - t + 1$. The number of time periods by which movement requirement r would miss the required delivery date if it arrived on day t (Derived Data).

24

AL(r) = Number of days movement request r may be delivered late. This parameter has two purposes: It defines a constraint and it assigns priorities to MRs.

Supply(r,i) = Quantity of movement requirement r (stons) provided at POE i.

Demand(r,j) = Quantity of movement requirement r (stons) required at POE j.

$$IS(r,t) = \begin{cases} 1 & \text{if } t = ALD(r) \\ 0 & \text{otherwise} \end{cases}$$

$$ID(r,t) = \begin{cases} 1 & \text{if } t = MIN(RDD(r) + AL(r), NDAYS) \\ 0 & \text{otherwise} \end{cases}$$

Lift Asset Data:

CAP(a) = Lift capacity (stons) for a single lift on asset a.

Q(a,t) = The number of type a assets available during time period t.

UR(a) = Utilization rate (percent of time period available) for asset type a.

AC(a,i,j) = Cycle time (time periods) for asset type a to complete a round trip between POEs i and j. This time includes loading, refu·'·ng, and offloading.

TT(a,i,j) = Travel time (time periods) for asset type a to complete a single trip from POE i to POD j. TT is rounded up to the next time period. This precludes a movement requirement from making two legs of a trip in one time period. TT = 1 + CEIL (AC(a,i,j)/2)

SHIPIT = Time period multiple on which ship arcs may be used. (See Paragraph C.1 in this chapter for a description of SHIPIT and its use.)

C(a) = Cost factor. C(a) is a scaling factor used to rank order the cost for using various asset types. C(a) would be high for airlift assets and relatively low for other asset types.

25

Port Data:

    $E(i,t)$ = Throughput capacity (stons) of port i during time period t.

    NDAYS   = Number of time periods in the model.

Decision Variables:

    $X(r,a,i,j,t)$ = Amount of movement request r (stons) sent via asset a from POE i and arriving at POD j during time period t.

    $S(r,i,t)$ = Amount of movement requirement r (stons) remaining at POE/POD i at the end of time period t.

Model:

$$\text{MIN} \left[\sum_r \sum_a \sum_i \sum_j \sum_t x(r,a,i,j,t) * ((AC(a,i,j) + C(a)) + MD(r,t)) + \right.$$
$$a \in \text{aircraft}$$

$$\sum_r \sum_a \sum_i \sum_j \sum_t x(r,a,i,j,t) * ((AC(a,i,j) * C(a)) + MD(r,t)) +$$
$$a \in \text{sealift/overland}$$

$$\left. \sum_r \sum_a \sum_i \sum_j \sum_t x(r,a,i,j,t) * (AC(a,i,j) * C(a)) \right] \tag{1}$$
$$a = \text{elastic asset}$$

Subject to:

$$\sum_r \sum_i \sum_j x(r,a,i,j,t) \leq CAP(a) * Q(a,t) * UR(a)/AC(a,i,j) \tag{2}$$

for all a, t.

$$\sum_a \sum_j x(r,a,j,i,t) \quad - \sum_a \sum_j x(r,a,i,j,t+TT(a,i,j))$$

$$+ \; SUPPLY(r,i) * IS(r,t) \quad - \; DEMAND \; (r,i) * ID(r,t)$$

$$+ \; s(r,i,t-1) \quad\quad\quad - \; s(r,i,t) = 0 \tag{3}$$

for all r, i, t.

$$\sum_r \sum_a \sum_i x \; (r,a,i,j,t) \leq E(i,t) \tag{4}$$

for all j, t.

$$\sum_r \sum_a \sum_j x \; (r,a,i,j,t) \leq E(j,t) \tag{5}$$

for all i, t.

$$x(r,a,i,j,t) \geq 0 \tag{6}$$

for all r, a, i, j, t.

$$s(r,i,t) \geq 0 \tag{7}$$

for all r, i, t.

It is very important to recognize the form of the decision variables. $x(r,a,i,j,t)$ is not a discrete "plane load" of movement requirement r being shipped from i to j. $x(r,a,i,j,t)$ and $s(r,i,t)$ are continuous variables that represent a flow rate/time period of commodity r on asset a from i to j. The advantage of this representation is a greatly reduced number of variables. If asset a could cycle between i and j five times in one time period, then five discrete variables would be needed

27

instead of one flow rate variable. The use of discrete variables would also require an integer formulation. Integer programs are much more difficult to solve and would place a substantial restriction on the number of decision variables which could be incorporated into the problem.

There are, however, two disadvantages to utilizing flow rate variables. In a small problem with only one time period and only one asset, you may establish two, three, or even more small flows from several ports all around the world. Obviously, this solution could not be executed and would not be acceptable. As already explained, deployment problems are not small problems. In a larger problem, it is assumed that a planner or ship scheduler would have sufficient assets to reasonably accommodate the flow rates established by the solution.

The second disadvantage is due to the different cycle times associated with air and sealift assets. Usually, time periods are kept short in order to maintain a reasonable resolution on air assets and their flow rates. There are usually many planes associated with a deployment plan. This makes it easy to visualize how these assets could be dispersed to meet the demands of the flow rates that have been established. Relative to air assets, however, there is usually a very limited quantity of sealift available. Just as in the small problem hypothesized above, the ability of a scheduler to apportion actual assets against the many possible flow rates can be disconcerting to the model user. The flow rates are also acceptable for modeling planes because planes

would actually be making deliveries during each time period. Sealift, on the other hand, would only be making a few deliveries during occasional time periods. This representation of sealift is not realistic, as "boat loads" appear more like "pipes." A method of lumping these sealift flows called "spiking" is discussed in the next paragraph.

1. Equation (1)

The objective function is a multiobjective function. It can be broken down into two cost components: Delivery Cost (DC) and Shipping Cost (SC). The primary purpose of the objective function is to penalize deliveries as they vary from the required delivery dates. This penalty is assessed by the Delivery Cost component. The second purpose of the objective function, subsequent to the first, is to select the most cost-effective means of shipping the movement requirements. This cost is assessed by the Shipping Cost component. The complete objective function is the sum of these two cost components. Total cost (TC) = DC + SC.

a   Explanation of Delivery Cost

$$DC = x(r,a,i,j,t) * MD(r,t)$$

Delivery Cost is the product of x (r,a,i,j,t) (the quantity of a movement requirement r delivered during time period t) and MD (r,t). MD $(r,t) = |RDD(r) - t| + 1$ represents the number of days the delivery missed the required delivery date $(RDD(r))$.

b. Explanation of Shipping Cost

SC = x (r,a,i,j,t) * (AC (a,i,j) + C(a)) for all air assets

SC = x (r,a,i,j,t) * (AC (a,i,j) * C(a)) for all non-air assets

SC = x (r,a,i,j,t) * (AC (a,i,j) * C(a)) for the elastic asset

Shipping Cost is based on two factors. The first factor is cycle time. AC (a,i,j) is the cycle time required for asset a to complete a round trip between ports i and j. If a C141 cargo jet's cycle time is less than that of a C5 cargo jet. then the C141 is considered a cheaper asset to use. This method of differentiating asset cost is adequate as long as we are comparing cost of similar types of assets. On the other hand, it is not immediately applicable to comparing sealift assets. which have relatively long cycle times. with airlift assets. The second factor affecting Shipping Cost is C(a). C(a) accounts for these differences in cycle times. C(a) is a scaling factor which is used in the three equations for SC given above. Note that C(a) is added to AC (a,i,j) for all air assets and is multiplied times AC (a,i,j) for all non-air (sealift. trains, etc.) assets. While this algebraic manipulation may appear odd at first. it provides a straightforward means of "tuning" the optimal solution to meet the desired trade-offs between expensive airlift and the cheaper transportation alternatives. Figure 3-1 depicts. in a simplified manner. the cost relationships between delivery dates and delivery mode. In this example. C(a) for air assets has ben set to 3.0. and C(a) for non-air assets has been set to .001.
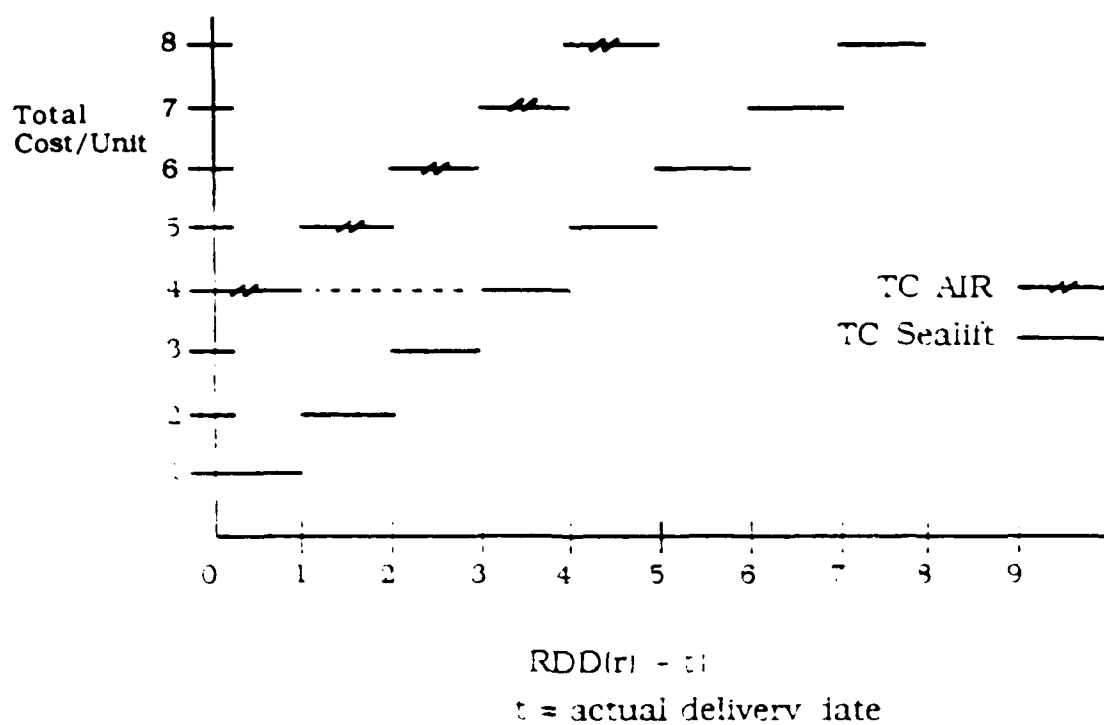
30

Figure 3-1

Cost Function

As Figure 3-1 shows, a delivery by train or ship could miss the required delivery date by approximately three days and still be cost effective when compared to an air shipment that arrives on the exact date required. This ability to sensibly balance alternate means of delivery (air vs. sealift) is the essential element of proper mode selection. Figure 3-1 also shows how $MD(r,t)$ quickly becomes the dominant factor in the Total Cost (TC) equation. This result is consistent with the primary purpose of the objective function.

31

c. Explanation of Elastic Shipping Cost

The purpose of the third SC equation is to provide *any* deployment problem with a feasible solution. This technique is often called an "elastic" or "soft" constraint [Ref. 12]. In this formulation, there is an unlimited number of hypothetical elastic assets available to transport any commodity r from i to j at an exorbitant price. C(a) for the elastic assets has been set to 1000. This very high relative cost factor insures that elastic assets are used only as a last resort. Without the elastic variables, the LP solver would terminate in an infeasible problem, yielding little or no information how to fix the infeasibility.

2. Equation (2)

This constraint ensures that the daily capacity of each lift asset is not exceeded. The product CAP(a) * Q(a,t) * UR(a) (capacity * quantity * utilization rate) determines the maximum quantity (stons) that can be transported in a single trip by asset type a. When this lift capacity is divided by the cycle time AC (a,i,j), we determine the maximum "flow rate" for each asset, from i to j, for one time period.

3. Equation (3)

This is the set of flow balance equations. They define for each movement request a single commodity network. The flow balance equations insure that the flow of every movement request r into node i is equal to the flow of MR r out of node i for any given time period. Figure 3-2 shows the flow components of movement requirement r into and out of node i during time period t. The problem of ensuring that supplies and demands are in balance is a simple one in the
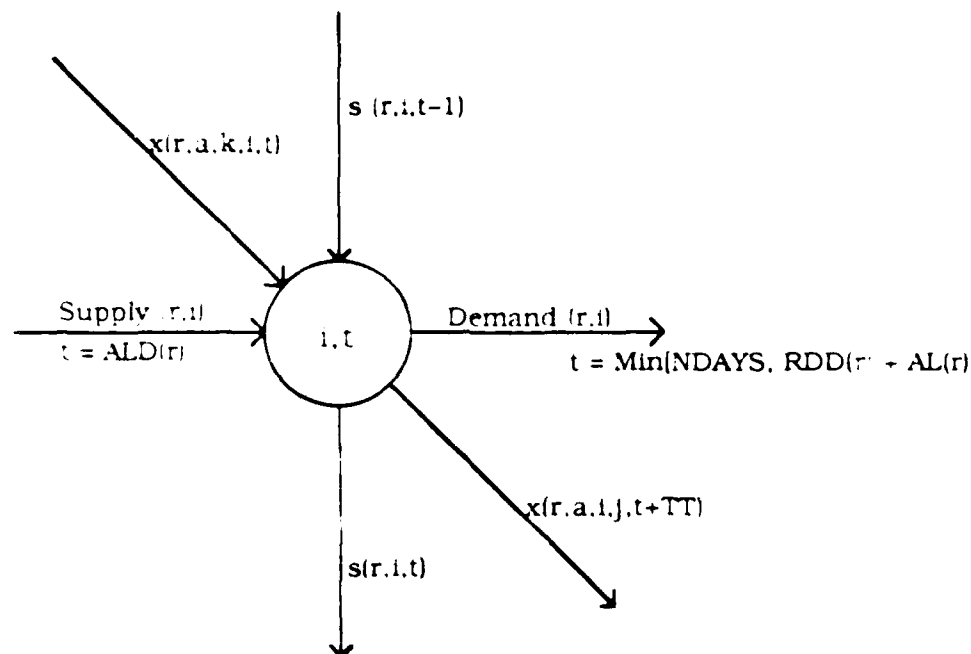
32

Figure 3-2

Flow Balance About a Node

deployment transportation problem. Since supply and demar.d. for a specific movement request. are prescribed quantities. supply will always equal demand. It is also known that the supply of r will enter the network during time period $t = ALD(R)$. and that movement requirement r will exit the network during time period $t = Min[NDAYS.RDD(r) + AL(r)])$.

4. Equations (4) and (5)

These constraints insure that the daily capacity of each port is not exceeded. The sum of all shipments made from POE i and to

33

POD j cannot exceed the quantities E(i,t) and E(j,t), respectively, during any time period t.

C. FORMULATION ENHANCEMENTS

1. Spiking Sealift

To provide for a more realistic model of sealift, the set of possible shipping flows must be modified. A simple example is provided to show this need. Consider the requirement to transport 10,000 stons from i to j on a sealift asset (type a). If the cycle time for asset a were 10 days (AC (a,i,j) = 10), the 10,000 stons would appear at j as 10 daily deliveries of 1,000 stons each. As we have previously discussed, this representation is not realistic and makes sea deliveries appear more like pipes than ships.

Since the preferred integer solution to a problem of this size cannot be obtained, a technique called "spiking" sealift was developed. This technique is used to consolidate the deliveries into a reduced subset of time periods in which sealift could be used. This feature is controlled by the variable SHIPIT. If SHIPIT is set to five, then sealift deliveries can be made only during every fifth time period. This would result in the previous example of 10,000 stons being delivered in two shipments instead of 10.

The technique of "spiking" the sealift flows must be used carefully and the modeler must be aware of its shortcomings. The most notable shortcoming is the loss of solution flexibility. The final solution can only provide sealift deliveries on every fifth day, even if it were physically possible and more cost-effective to make a delivery

34

during another period. Although a more realistic representation of sealift is the primary purpose of spiking, it also serves to reduce the number of sealift decision variables which must be considered. Figure 3-3 shows both effects of spiking sealift flows.

2. Prioritizing Movement Requirements

A second enhancement is that of prioritizing the delivery of movement requirements. Movement requirement priorities are established by adjusting the allowable late factor (AL(r)) for each MR. If, for example, we want to place a high priority on movement requirement r, we can set $AL(r) = 0$. No solution will allow MR(r) to be delivered late. A lower priority results when higher values of AL(r) are assigned. The last day in the problem (NDAYS) will of course override the allowable late factor's ability to let MRs be delivered late. A larger value for AL(r) will also provide the model with a more flexible number of time periods in which to find a feasible solution. The cost of this flexibility is the addition of more decision variables. The effects of alternate AL(r) values is depicted in Figure 3-4.
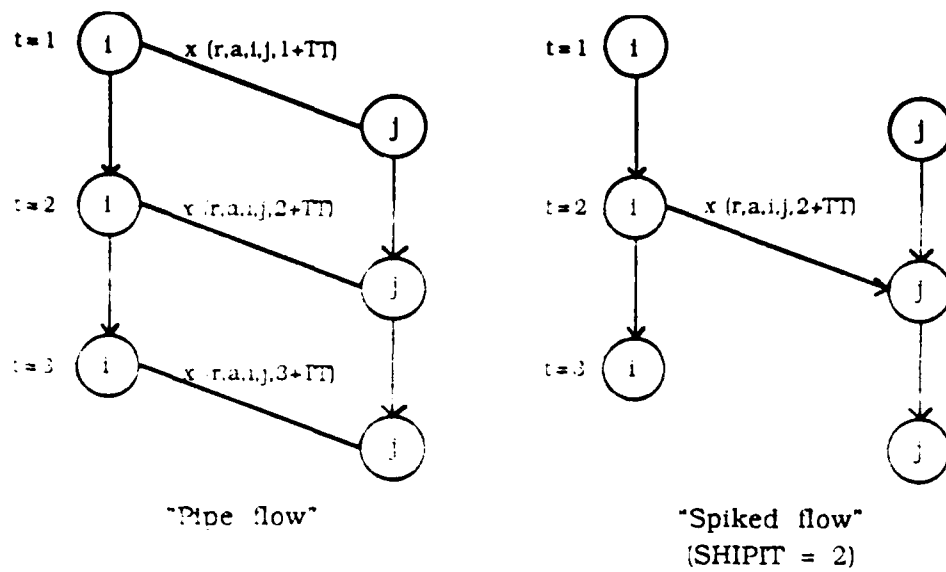
"Pipe flow"

"Spiked flow"
(SHIPIT = 2)

Figure 3-3

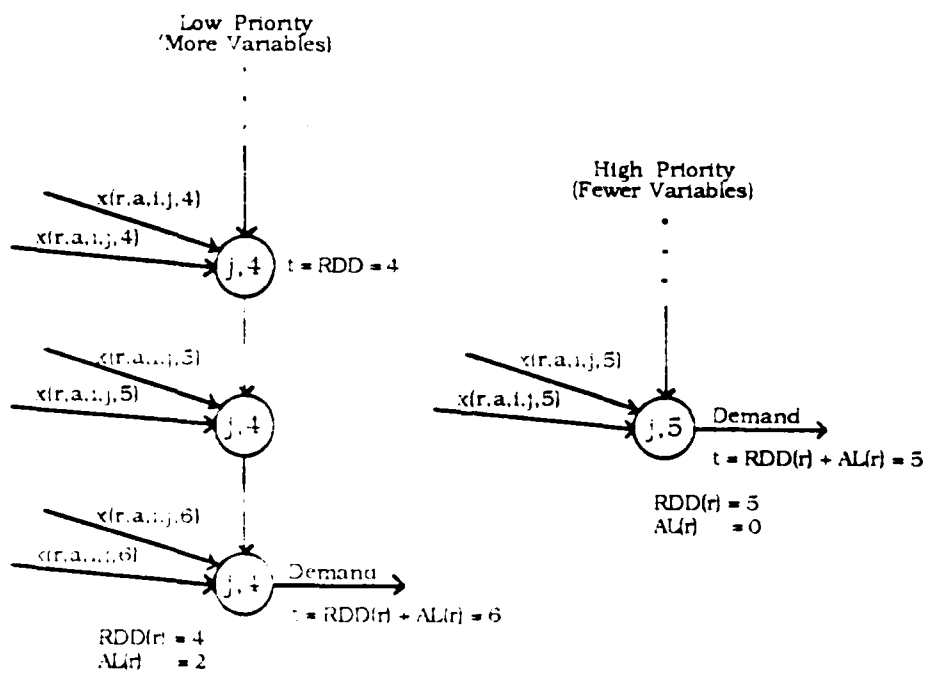**"Spiking" Sealift Flows**

Figure 3-4

Prioritized Deliveries

IV. REDUCING THE NUMBER OF DECISION VARIABLES

As previously defined, the optimization model includes two types of decision variables: Shipping variables, x (r,a,i,j,t) ε X and Inventory variables s (r,i,t) ε S. The domain of X potentially contains $r*a*i*j*t$ variables. Likewise, S would include $r*i*t$ variables. In the example already mentioned with (r,a,i,j,t) = $(500,10,40,40,90)$, there are in excess of 720 million decision variables and 2 million constraints. These dimensions for (r,a,i,j,t) correspond to a medium- to large-sized deployment problem. Even larger numbers may be encountered in practice. Clearly, a straightforward approach to a problem of this size would not be feasible.

This chapter presents the development of an algorithm which is designed to greatly reduce the number of variables found in deployment problems like the one described above. Since a decision variable is analogous to an "arc" in a directed graph, the algorithm to be developed has been entitled the Arc Reduction Algorithm (ARA).

This chapter is organized into three sections. The first section presents the design of the ARA along with the criteria it uses for reducing the number of decision variables. The Arc Reduction Algorithm is presented in pseudo code in Section B. Section C explains how the Arc Reduction Algorithm and deployment problem input data are both used to generate the final (reduced) set of decision variables that represent the transportation network.

A. DESIGN AND CRITERIA FOR AN ARC REDUCTION ALGORITHM

1. Design

A "path" from node s to node u may be defined as a "sequence of arcs" P_{su} = {(s,a), (a,b), (b,c), ..., (e,f), (f,g), (g,u)} [Ref. 10:p. 406]. Each of the arcs in path P_{su} corresponds to an element of X or S, the two sets of decision variables.

In the context of a deployment problem, Figure 4-1 shows an example of two possible paths for a movement requirement. Path 1 contains five decision variables (arcs) but path 2 needs only one decision variable. Any solution to a deployment problem must provide at least one path for each movement request r from node POE(r) to node POD(r).

The design for the arc reducing algorithm is to attempt a deliberate search for at least one "good" path from s = POE(r) to u = POD(r) for each movement requirement r. Then, decision variables x (r,a,i,j,t) or s (r,i,t) are retained for the optimization only if they have been associated with some good path. This search procedure is performed independently for each movement request r, one at a time. A common approach to organizing such a deliberate search is the "Depth-First Search" (DFS) algorithm [Ref. 14].

The purpose of the Depth-First Search (DFS) is to efficiently visit the vertices and arcs of a directed graph in a systematic, step-by-step (arc-by-arc) fashion. The technique is called depth-first because it continues searching deeper (in the direction away from the starting node) for as long as possible [Ref. 14:pp. 215-216]. Figure 4-2 shows

38

Path #1 – X (Train, Ft. Benning, Charleston, Day 2)
S (Charleston, Day 2)
S (Charleston, Day 3)
X (RORO , Charleston, Rotterdam, Day 14)
X (Train, Rotterdam, Frankfurt, Day 16)

Path #2 – X (C141, Ft. Benning, Frankfurt, Day 16)

Key: X = Shipping Variable
S = Storage Variable

Figure 4–1

Possible Shipping Paths

Ft. Benning, GA

Charleston, SC

RORO

Rotterdam

Frankfurt
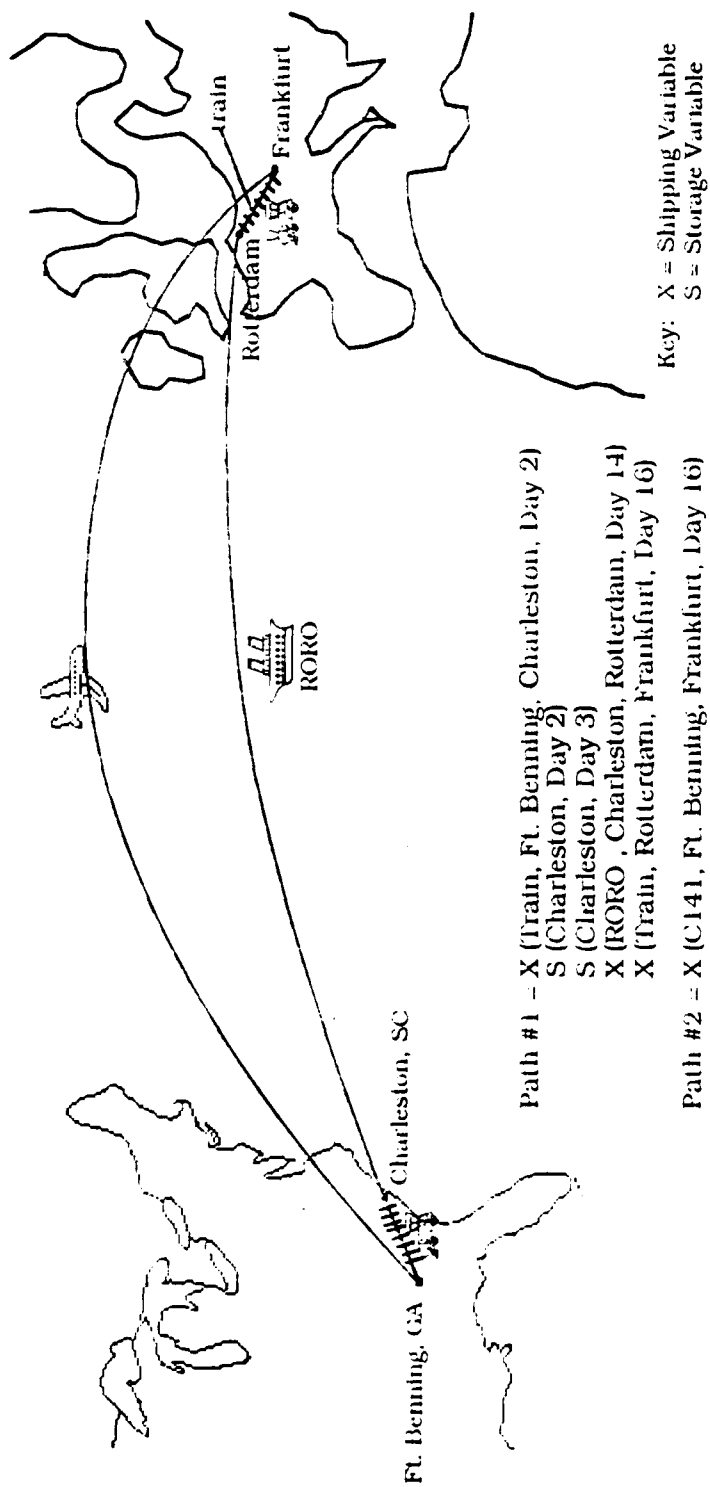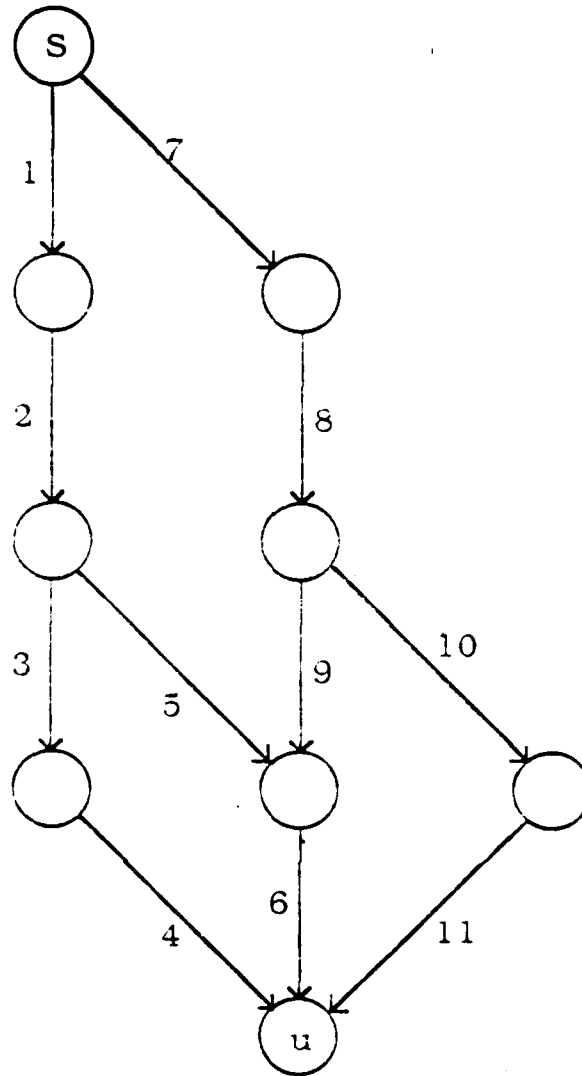
Train

Figure 4-2

Acyclic Graph After DFS

an acyclic directed graph whose arcs have been labeled in the
sequence they would be visited during a DFS.

The ARA presented in the next section utilizes the DFS tech-
nique to seek out paths from s to u. Once a complete path has been
found, each of the decision variables contained in the path will be

40

identified and retained for the optimization. The ARA algorithm does not need or attempt to find *all* paths from s to u. Such an exhaustive search would not provide any additional information.

The ARA utilizes this fact to its advantage in an effort to increase its efficiency. Let (i,j) represent an arbitrary arc. The ARA will generate every possible arc leading into (i,j). Two conditions may exist when the DFS reaches (i,j). Either (i,j) has already been included in a complete path from s to u or (i,j) has never been included in a complete path. If (i,j) has not been included in a complete path, then the DFS will continue deeper in an effort to do so. The search along this path, which includes (i,j), may or may not ever complete a path to u. If it does, (i,j) will be retained for the optimization. If, when the DFS reaches (i,j), (i,j) has already been included in a complete path, then the DFS does not have to go any further. The path leading up to (i,j) can be combined with the previously established path from (i,j) to u to form a new complete path. The fact that we do not have to proceed deeper on completed paths contributes greatly to the algorithm's efficiency.

If, after approaching (i,j) from every possible origin, we have yet to include (i,j) in a complete path, then we can conclude that (i,j) should not be retained for the optimization. It is important to note that (i,j) was given every opportunity to be included in a complete path. This distinction becomes a requirement when we assert that we have not inadvertently precluded any decision variable from a possible optimal solution.

41

This assertion is an important feature of the ARA. In other words, if the ARA has accounted for every required decision variable, then every possible path from s to u has also been found. If every possible path has been found, then the optimal solution has not been affected by our reducing the number of decision variables which the optimizer may consider.

2. Criteria For Reducing the Set of Arcs (Decision Variables)

While the DFS procedure will systematically provide us with the requisite paths from s to u, the following set of "acceptable criteria" will determine if the path P_{su} is a "good path":

a. P_{su} must insure that movement requirement r was not picked up before its available-to-load date (ALD(r)), or delivered after its required delivery date (RDD(r)).

b. The aircraft arcs belonging to P_{su} should not exceed the direct route distance from s to u by more than a reasonable factor. In the current model, this factor is an adjustable parameter which defaults to 1.50.

c. Along path P_{su}, the number of times an aircraft is loaded or unloaded should not be excessive. In the current model, this factor is an adjustable parameter which defaults to 3. (It can be different for cargo and passengers).

d. Path P_{su} should not return to the POE at any point after it has departed the POD.

e. Path P_{su} should not leave the POD once it has found it.

B. ARC REDUCTION ALGORITHM (ARA)

1. Pseudo Code

The following is a simplified set of variables and data used by a pseudo code to represent the arc reduction algorithm.

Define:

Sets:

MR = set of all movement requests, indexed by r.
A = set of all asset types, indexed by a.
V = set of all network vertices (ports), indexed by v.
T = set of all time periods, indexed by t.
E = set of all edges (decision variables).
GA = set of "good arcs" (arcs that have been included in one or more paths P_{su} and retained for optimization).

Data:

s = Initial vertex (POE(r)); $s \in V$ and s = POE(r) for current r.

u = Destination vertex (POD(r)); $u \in V$ and u = POE(r) for current r.

TT(a.i.j) = Travel time from i to j on asset a: TT = 1 if i = j (represents storage variables).

ALD(r) = First time period r will be available to load.

LDAY(r) = Last time period r may be delivered.

Variables:

i = Tail of the current arc; $i \in V$.

j = Head of the current arc; $j \in V$.

(i,j) = arc connecting i to j; $(i,j) \in E$. This arc or decision variable may represent either an inventory arc— s(r.i.t), or a shipping arc— x(r.a.i.j.t).

P_{su} = Path from s to u (represents a complete path, not necessarily unique, from POE(r) to POD(r).

P_{sj} = Path from s to j, s.t. $P_{sj} = P_{si} \cup (i,j)$ (represents the path being built).

P_{iu} = Path from i to u, s.t. $P_{iu} = (i,j) \cup P_{ju}$ (represents a previously established path from the current arc (i,j) to the destination at u).

d = Depth of P_{sj} (depth = number of arcs in the path).

Pred(d.k) = Predecessor array (a vector of length k, for each d, which contains the information required to "backtrack" along P_{su} or P_{sj}).

Functions:

ACCEPTABLE $(P_{si} \cup (i,j))$ = TRUE; if the new path, $P_{si} \cup (i,j)$, meets the criteria specified earlier in Paragraph A.1. of this chapter.

43

Arc Reduction Algorithm:

input: $G = (V,E)$, a directed acyclic graph (not necessarily connected); POE(r) the initial vertex; POD(r) the destination vertex; TT(a,i,j), edge length cost/travel time.

output: GA: List of all arcs which were at one point included in any P_{su}.

```
begin
  for r ε MR
  begin
     d=0, s=POE(r), u=POD(r), t=ALD(r), ?=LDAYS(r)
     P_su = Ø, P_sj = Ø, P_ju = Ø.
     i = s
Next-V: for v ε MV
         begin
           j = v
           for i ε A
           begin
             if ACCEPTABLE[P_sj U (i,j)]] add arc (i,j) to P_su
             begin
               P_sj = P_sj U (i,j)
               i = j
               t = t + TT
               d = d + 1
               if i = u or P_sj U P_iu = P_su    (path from s to u is complete)
               begin
                 GA = GA U (i,j) for ∀ (i,j) ε P_su
                 (i,j) = PRED(d,k)
                 d = d - 1                        (backtrack on P_su)
                 go to Next-v
               end
             else
             end
           end
         end
         if d > 0
           begin
             (i,j) = Pred(d,k)
             d = d - 1                          (backtrack on P_sj or P_su)
             go to Next-V
           end
         else
         end
  end
end
```

2. Data Structure Used to Implement Arc Reduction Algorithm

a. Preordered Traversal of the Graph

Because the deployment problem is very structured, it is possible to systematically generate the arcs required in the depth first search only as needed. The ARA above depicts the cyclic structure of this search and how the arcs are generated. This arc generation technique is used instead of a hierarchical adjacency list.

While this technique may appear crude on the surface, it may be as good as any other alternative to preordering the search. Clearly, if there were only one commodity to be shipped, then an adjacency list could be generated that would be much more efficient than the iterative routine being used here. It would, however, be a difficult problem to develop a single adjacency list that would account for different POEs, PODs, ALDs, and RDDs for each commodity type or movement requirement. The simpler alternative, which would generate a new adjacency list for each movement request, would certainly prove to be more time-consuming than our iterative generation technique.

b. Storage of Good and Bad Arcs

Each time the ARA attempts to add an arc to the path it is building, it is necessary to classify that arc as being good or bad. Three operations/situations complicate this task.

The first problem is encountered when you immediately classify an arc as being good each time a "step forward" is taken. Obviously, if the path never reaches the correct destination, then time

45

must be taken to remove and reclassify one or more arcs as we back-track along the path. A common technique used to maintain this type of changing list is the "last-in-first-out" (LIFO) stack [Ref. 14:p. 215]. The ARA saves a portion of the time that would be required to "pop" (remove) arcs from this stack by not immediately classifying each for-ward step taken as a good arc. Instead, these arcs are maintained in the predecessor array and "flagged" when the path actually reaches the correct destination. All the arcs that have been flagged are then added to the "good" list of arcs. This procedure eliminates the requirement to ever remove an arc from the good list of decision variables.

The second complication occurs when the current path being built determines that an arc that was previously classified as bad may now be acceptable. Neither a LIFO or a FIFO stack would help in correcting this reclassification. The arc in question could at this time be anywhere in the stack. The simplest means of removing this arc from the bad stack is to delete it and replace it with the last member of the stack. This procedure was used and should be faster than updating the pointer to every arc below it in the stack.

The third problem is also related to managing the stacks of good and bad arcs. Each time the ARA attempts to take a forward step on an arc, it must search the two stacks to determine if they have been "visited" before and how they have been classified (good or bad). The stack of good arcs also has to be searched during each back-tracking step. Since these stacks will be searched $o(|V|^2 * |A| * |R|$

* |T|) times. the program runtime will be influenced greatly by the length of these stacks. We are fortunate that, once the search has terminated for each movement request. we can store the good arcs in a separate file. All the stacks can then be purged prior to starting the next search iteration. If only two stacks were maintained, we would still be greatly influenced by their length. To reduce the lengths of these stacks, a "bucketing" (classification) technique was used to disperse both the good and bad arcs into many smaller stacks [Ref. 14:p. 122]. A good and bad stack was created for each node in the network. Since the ARA always knows its location in the network. it can directly access the appropriate stack.

One additional step was taken to save storage space for these stacks. Since the inventory arcs have only two components versus the shipping arcs' four, space can be saved by further breaking down the stacks into these two categories. By doing so we have also once again reduced the length of each stack. As a result of this stack partitioning scheme, we must keep track of 4 * |V| individual stacks and stack counters. This data storage technique assists greatly in reducing program run time since any stack we must search will certainly be a relatively short tack.

C. NETWORK GENERATION

The network of transportation links is established in two phases. During the first phase, the modeler constructs the "physical" network by means of a "linking" array. This array is the AC (a.i.j) data array described in the mathematical formulation. The AC (a.i.j) linking array

47

serves two purposes in this formulation. The first purpose is to identify those transportation links (physical network) which will be allowed to exist in the model. The second purpose, as was described in the formulation, is to establish the cost for asset a to transport movement requirement r from i to j. The linking array has n columns, one for each port, and n sets of a+1 rows (a = number of asset types). The ath row in the nth set is the vector of cycle time (cost) for asset a as it leaves the nth port. The (a+1)st row in each set is a normalizing distance between port i and port j. Any ratio scale, such as miles between i and j, can be used for this purpose.

Positive AC (a,i,j) values establish an actual transportation link for asset a between ports i and j. An example such as AC (C141, New York, Frankfurt) = 1.4 would indicate that the optimizer should consider transporting movement request on C141 cargo jets from New York to Frankfurt at a cost of 1.4 time periods. The array value for AC (Container Ship, New York, Denver) would on the other hand be set to zero.

When constructing this linking array, the modeler can use one of two approaches. In the first approach, which may be applicable to planning in a crisis mode, the modeler could simply extract his AC (a,i,j) linking array from a data base. In this case, the array would have a positive value for every a,i,j) combination that is actually possible. The advantage to this approach is that the modeler does not have to have much detailed knowledge about the plan and it would require

48

very little time. The disadvantage is that the optimizer must now consider a great many more transportation links than may be required.

In the second approach to creating the AC (a.i.j) linking array, the modeler is more selective in what links are established. When more time is available and the modeler is more familiar with the deployment plan, the AC (a.i.j) array will contain fewer non-zero elements. This will make the size of the coefficient matrix smaller and the job of the optimizer that much easier.

Phase two in creating the transportation network is a much more complicated process. During this phase, the Arc Reduction Algorithm is used to extract from the physical network created in phase one only those links that have been classified as acceptable.

The second phase of generating the network is accomplished during each major iteration of the Arc Reduction Algorithm. A major iteration of the ARA is complete when every attempt has been made to associate each decision variable with a "good" path from POE(r) to POD(r) for a particular movement request r.

At this point in the algorithm, a list of good variables which are to be used by the optimizer has been established. The next step is to insure that each of these variables, along with their corresponding coefficients and constraints, is placed into the Mathematical Programming System format (MPS). This is accomplished by calling each of three subroutines (ROWS, COL, RHS) prior to purging the list of variables and restarting the algorithm for the next movement request.

V. **RESULTS AND CONCLUSIONS**

This chapter describes each of the components that have been developed for the SCOPE-NPS model and how they operate together as a system. The two other systems used for development and implementation in this study— GAMS/MINOS and the optimizer MPS III— are also briefly described. The third section of this chapter presents the results of the testing and validation phase of this study. The final two sections of this chapter present some conclusions and recommendations for future developments.

A. **MODEL COMPONENTS**

The SCOPE-NPS model consists of three components: the Data Input Array, the Arc Reduction Algorithm, and the Matrix Generator. The ARA reads the Data Input Array and begins the iterative process of finding all the good paths for each movement requirement. At the end of each iteration, the Matrix Generator is called (as a subroutine) and the problem is converted to an MPS formatted file. When the paths for each movement requirement have been found through this iterative process, the work of the SCOPE-NPS is complete. The SCOPE-NPS output file is in the MPS format and can be solved by any linear programming system that reads MPS files. The solver selected to support the SCOPE-NPS was the MPS III Mathematical Programming System developed by Ketron Management Science, Inc., for use on IBM mainframe computers [Ref. 15].

The SCOPE-NPS model formulation was initially developed and implemented utilizing GAMS/MINOS. GAMS/MINOS is a software package consisting of GAMS, the General Algebraic Modeling System, and MINOS, the Modular In-core Nonlinear Optimizing System [Ref. 16]. The GAMS language allows the modeler to enter his LP/NLP/MIP model in an algebraic form. The user must specify each of the sets, parameters, and variables for the model, but he only needs to enter a single statement in GAMS language for each type of constraint or relationship. The GAMS compiler will, in turn, generate the entire set of required equations when they are needed. This arrangement frees the modeler from the tedious work which is required to develop and revise a matrix generator. Although the formulation was developed on an IBM PC, final model testing was conducted on the IBM 3033 mainframe version of GAMS/MINOS.

The ARA and Matrix Generator were developed and implemented on the IBM 3033 AP computer operating under the CMS operating system. The ARA is written in approximately 600 lines ANSI FORTRAN 77 and compiled by the IBM VS FORTRAN compiler. An additional 400 lines of FORTRAN 77 code was required to program the three Matrix Generator subroutines.

1.   Data Input Array (DIA)

The DIA is a formatted array which can be divided into two parts. The first part provides information concerning the size of the deployment problem. The information contained in this section includes: number of movement requests, number of asset types,

51

number of ports, number of days in the problem, number of aircraft types, and number of boat types.

The second portion of the DIA contains the following parameters and data list: air transport cost coefficient, land or sea cost coefficients, maximum number of planes allowed in a single path, fraction of a direct route planes may fly on a single path, time period multiple on which ships may be used, movement requirement data, port capacity data, lift asset data, and cycle time cost (AC (a,i,j)).

2. Arc Reduction Algorithm

The ARA reads the problem size specifications and physical network data from the Data Input Array. As discussed in Chapter IV, the ARA then proceeds to identify a reduced set of decision variables which are retained for the optimizer. The ARA is run one time for every movement requirement in the problem. At the end of each run, the Matrix Generator's subroutines are called to convert the new set of variables into the MPS format.

The ARA is the key element which allows the SCOPE-NPS model to solve realistically sized deployment plans. The following ARA test run results in Table 5-1 demonstrate the ability of the ARA to reduce the size of a deployment problem.

TABLE 5-1

RESULTS OF THE ARA

| Problem Size | | | | | Number of Decision Variables | | Percent | CPU |
r	a	i	j	t	Before Reduction	After Reduction	Reduction	TIME (secs)
1	3	4	4	8	416	32	92 %	.04
5	4	9	9	6	9,990	176	98 %	1.19
20	4	9	9	50	333,000	2,200	99.5 %	19.25
90	9	20	20	90	35,461,800	11,150	99.99%	296.56

3. Matrix Generator

The matrix generator reads the list of "good" variables that are supplied by the ARA and converts them in accordance with the mathematical formulation to the MPS format (see MPS format [Ref. 17]). The MPS format has long been a standard format in which linear programming problems are input to solvers.

B. MODEL TEST RUNS

A series of three tests were used to test the SCOPE-NPS model performance. The purpose of TEST #1 was to verify the performance of the Arc Reduction Algorithm. The purpose of TEST #2 was to verify on a small deployment problem the performance of the complete SCOPE-NPS model (ARA, the Matrix Generator) and the MPS III optimizer. TEST #3 was designed to demonstrate the ability of SCOPE-NPS and MPS III to solve a realistic (medium) sized deployment problem.

Since the SCOPE-GT model was the first model applied to the joint military transportation problem, "there is no validated

53

benchmark solution data for comparison and validation" [Ref. 2:pp. 2-3]. Because a benchmark deployment problem does not exist, the five deployment plans used to test the SCOPE-NPS model were all designed during this study. In each of the first four deployment test plans, each movement request was specifically designed to test for the presence of a particular solution attribute. These movement requests were carefully matched to a simple physical network in order to provide obvious good or bad examples of solution behavior.

The following list is a small sample of the attributes which were to be tested:

- Would the Arc Reduction Algorithm adhere to the rules for selecting decision variables?

- Would port and asset capacities be adhered to?

- Would movement requirements be picked up at the correct location on the correct date?

- Would movement requirements be delivered to the correct destination on or about the required delivery date?

- If given the choice between two paths from the POE(r) to the POD(r), would the solution select the cheaper alternative? (i.e., if time were available, would the solution select a sealift movement over an airlift movement?)

- Would the solution correctly utilize the "super tanker" (elastic constraint) to maintain a feasible solution?

1. TEST #1

The purpose of this test was to test the Arc Reduction Algorithm's ability to identify a set of "good" paths in accordance with the rules established in Chapter IV. Three small networks were designed for the express purpose of testing each of the appropriate rules. In

each test run, all the acceptable paths were correctly identified. Each arc component which had been a member of a good path was accounted for and placed into the list of decision variables to be retained for the optimizer. The ARA also identified correctly each of the paths that had been designed to violate one of the acceptable path rules. These paths, along with their arc components, were never included in the final set of decision variables.

2. TEST #2

TEST #2 was the first test of the complete SCOPE-NPS model. The purpose of this test was to insure that an optimal solution possessed the correct attributes as required by the original model description. TEST #2 was accomplished in two phases. Phase one of this test was to insure that the proper solution attributes were being produced by the model. This phase was conducted on the GAMS/MINOS optimizer. During phase two of this test, the SCOPE-NPS model and MPS III solver were expected to duplicate the optimal solution from the GAMS/MINOS model. The deployment plan used during this test required that five movement requests be transported among nine ports and delivered according to a prescribed six-day schedule.

In phase one, the mathematical formulation given in Chapter III was developed and refined. This phase of testing was the most important of all three tests. The ability to solve larger problems would be of little use of we were not confident that the solutions being provided on this small scale were not correct.

A major portion of this testing phase was devoted to a sensitivity analysis. The purpose of this analysis was to insure that proper solutions would be obtained as the problem situations changed. Initially. decision "break points" were identified for each required solution attribute. For example. the following situation would create a decision break point for a particular movement request r. Suppose that a RORO cargo ship has an eight–day travel time from POE(r) to POD(r). If the required delivery date (RDD(r)) for this requirement is prior to day 8. then a feasible solution would require an aircraft to get it there in time. If the RDD(r) was after day 8. then the solution should allow for the cheaper RORO cargo ship to make the delivery. If there are no other conflicting constraints. proper model behavior can be tested by adjusting the RDD(r) to both sides of this decision break point.

This procedure was continued until proper solution behavior was obtained on both sides of each model attribute or decision "break point" of concern. When the formulation had proven that it could flexibly provide acceptable solutions to the test deployment problem. phase one of TEST #2 was concluded.

The purpose of phase two of this test was to validate the MPS III solution to the SCOPE–NPS model. Since we already had a "benchmark" solution from phase one of this test. it would be easy to validate the SCOPE–NPS model. When the MPS III solution proved to be the same as the GAMS/MINOS solution. both the ARA and the Matrix Generator were shown to be functioning properly.

Although an identical solution was obtained during phase two, a major improvement resulted from the reduced number of decision variables that the SCOPE-NPS model provided to the MPS III optimizer. Initially, the number of decision variables in this deployment plan was approximately 9,990. Utilizing the "such that" ($) control operator in GAMS, the number of decision variables was reduced to approximately 1,800 [Ref. 18]. The ARA reduced the initial set of 9,990 decision variables down to 176.

The fact that all the variables in the optimal solution were included in the reduced set of 176 variables was a very important developmental milestone. It verified that a 98-percent reduction in the number of decision variables being considered could be "intelligently" accomplished without affecting the optimal solution.

3. TEST #3

The purpose of TEST #3 was to demonstrate the ability of the SCOPE-NPS model and MPS III optimizer to solve a realistic, medium-sized deployment problem. The deployment plan designed for this test was given the name "OPLAN TEST-3." Many of the OPLAN characteristics concerning the movement requirements, asset and port allocations, and travel times were extracted from the JDA test deployment plan "MODELD 123DF02."

Briefly, OPLAN TEST-3 required that 90 movement requests be transported among 22 ports according to a 90-day schedule. Thirteen of the ports (eight airports and five seaports) were located in the US and the remaining nine ports were in Europe (four airports and

57

five seaports). Nine types of lift assets were also provided. They included four types of cargo planes: C130, C141, C5, and LRWB (long-range wide-body, DC-10 or Boeing 747); four types of sealift: RORO, Breakbulk, Container (fast), and Container (slow); and a "train." The purpose of the train was to represent all "surface" shipments: trains, trucks, and road marches. A straightforward approach to solving this problem would require the optimizer to consider a set of 35,461,800 decision variables.

Figure 5-1 summarizes the required deliveries scheduled in OPLAN TEST-3. This schedule is typical of a deliberate deployment. During the first 20-25 days, there is a gradual build-up of forces. This build-up is followed by the arrival of the main deployment body. This phase of a deployment is, of course, the most resource intensive. Following the main body deployment there is a reduced but steady stream of movement requirements designed to reinforce and sustain the deployed forces.

SCOPE-NPS and MPS III solution results for OPLAN TEST-3: The SCOPE-NPS took approximately 296 seconds of CPU time to reduce the set of variables and to create the MPS file. The MPS III optimizer required 95 seconds of CPU time to provide the optimal solution. The solution to OPLAN TEST-3 was obtained in less than one percent of the time required by the SCOPE-GT model to solve a similar problem. In all fairness, it must be said that the SCOPE-GT model does much more than the model presented in this study. Its formulation considers more aspects of the deployment problem and it
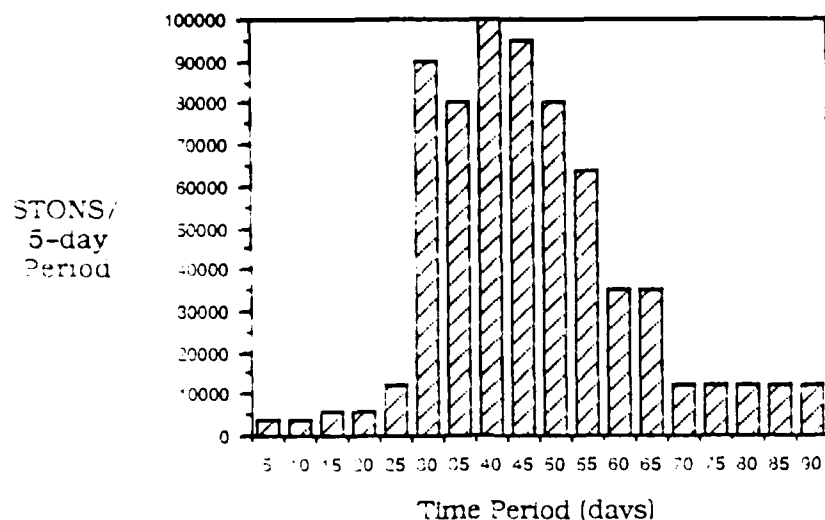
58

STONS/
5-day
Period

Time Period (days)

Figure 5-1

Delivery Schedule
OPLAN TEST-3

creates its own data input file directly from OPLAN records. The
SCOPE-NPS model does, however, provide a more reasonable solution
in a fraction of the time.

MPS III statistics revealed that 36 percent of available mem-
ory had been used to solve this problem. Based on these statistics,
there is an obvious potential to solve larger and more detailed deploy-
ment problems.

Most of the following solution results are portrayed graphi-
cally in Figure 5-2.

• The ARA reduced the number of decision variables to be consid-
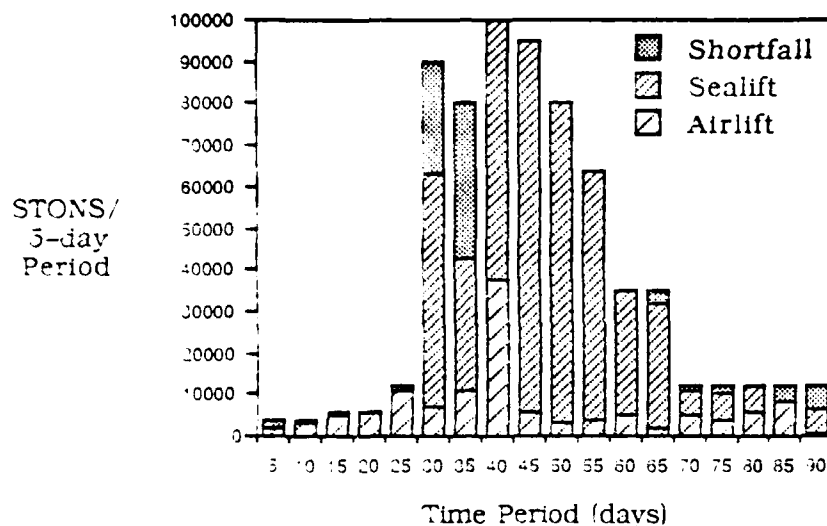 ered from 35,461,800 down to 11,150.

59

Figure 5-2

Optimal Delivery Schedule
OPLAN TEST-3

- When time was available. the model consistently selected the sealift mode of transporting each movement requirement.

- Approximately 69 percent of the entire deployment was transported across the Atlantic Ocean by sealift.

- Table 5-2 displays a portion of the sealift deliveries made during the period day 30 to day 36. The following sample demonstrates how the "spiked sealift" appeared in the solution.

The "spiking" technique seems to be a marked improvement over the continuous "pipe flow" of sealift experienced in a SCOPE-GT solution.

TABLE 5-2

"SPIKED" SEALIFT OPTIMAL SCHEDULE

Movement Request	Day of Delivery	POE	POD	Quantity (stons)	Ship Type
26	30	Norfolk	Rotterdam	16,000	Breakbulk
27	30	Norfolk	Hamburg	16,000	Breakbulk
28	36	Houston	Hamburg	10,000	RORO
29	36	Houston	Hamburg	10,000	RORO
30	36	Houston	Zeebrugge	2,000	RORO
30	36	Houston	Zeebrugge	18,000	CC-(Fast)

- The solution identified the following shortcomings in the deployment plan:

  1. During the initial buildup phase (days 1-30), there was a 16 percent shortfall in deliveries. Due to the compressed time frame of the buildup phase, this shortfall could only be corrected with an increase in airlift assets.

  2. The most significant shortfall in deliveries occurred during the first 10 days of the main deployment (days 30-40). During this period, there was a delivery shortfall of 64,000 stons. This represents 75 percent of the total shortfall.

  3. During the last 20 days (days 45-65) of the main deployment, there was an excess of airlift assets assigned to this deployment. This large allocation of aircraft was very beneficial from days 40 to 45 but was excessive once adequate sealift assets had arrived on about day 45.

  4. When the deployment moved into the sustainment phase, airlift assets were scaled down too much. This shortage resulted in a 76-percent shortfall in high-priority shipments during the last 25 days of the deployment.

61

C. RECOMMENDATIONS FOR FUTURE STUDY

1. The Arc Reduction Algorithm can be modified into a "conservative" heuristic so as to further reduce the set of decision variables. The Arc Reduction Algorithm in its current form is guaranteed not to affect the optimal solution. For the purposes of evaluating a deployment plan at the "closure planning" level. this guarantee may be too restrictive. A simple example will make this point clear. A five-ton movement requirement does not need *every* "good" path that the ARA will find for it. It may only need one of the several dozen that may be available. On the other hand. a 25.000-ton movement request may need access to every path available. The solution for the five-ton MR may be to select only two paths (one by air and one by sea) if they exist. Retaining at least two paths in this case would still enable the solver to look for the cheapest. yet acceptable. shipping alternative. This smaller set of decision variables may not be able to guarantee an optimal solution. but it will still enable the solver to answer questions such as. "Is this OPLAN transportation feasible?"

2. Provided that the heuristic described above has been developed. there may be an advantage to converting the current node-arc formulation to a "chain formulation" [Ref. 19]. The advantage to this formulation can be shown in a short example. Suppose only two paths existed to transport a MR from its origin to its destination. and that these two paths were formed by linking together a total of seven decision variables. A node-arc formulation must consider all seven decision variables. The chain formulation would only have two decision

variables, one for each path. Once again, the problem has been greatly reduced in size and the potential exists to solve even larger problems.

The work that would be required to develop this formulation would be minimal since the ARA (heuristic) has already collected the information necessary to construct the chain variables.

3. Develop an LP formulation which uses a "cascading" technique to solve the complete large problem in a series of smaller ones [Ref. 20]. When attempting to solve large deployment problems, the current LP formulation and MPS III solver are the weak links. While the Arc Reduction Algorithm may be able to reduce the size of these large problems, the MPS III solver is limited to only 16,384 constraint rows [Ref. 15:p. 2-1]. A cascading formulation may be used to keep the number of rows being considered at one time within this limit.

4. The SCOPE-NPS does a good job of selecting what should be transported by sealift. It does not, however, come close to providing a realistic representation of sealift. As this study had originally intended, a "hand-off" or merging technique should be developed to combine the SCOPE-NPS model solution with the "Ship Scheduler" developed by Captain Lally [Ref. 2]. .

5. There are many formulation attributes which could be added to the SCOPE-NPS to make it a more flexible and realistic model. Some of the following formulation enhancements should be considered for future development:

a Develop the formulation so that it will take into account the critical loading constraints of each different transportation asset.

63

When loading aircraft, a key factor is the weight (stons) of what is to be shipped. When loading ships, the key factor is usually volume. A change in the formulation would allow the flow balance constraints to incorporate the conversion of units (stons to mtons—measurement tons) at designated nodes (ports) so as to account for the capacity factor which is most important.

b. An alternative, yet simpler, enhancement to the generalization described above would be to develop a conversion factor for representing the capacity of ships in terms of stons. This technique would preclude the need to convert units and still realistically model the problem.

c. Asset usage needs to be modified. The current model will always use the biggest and fastest asset until its capacity is exceeded. The next best asset type will then be used until its capacity is exceeded, and so on. In order to conserve assets, the work load needs to be distributed more evenly among each of the assets available.

d. The current formulation is only capable of shipping dry cargo. With minor modifications, the formulation can be adopted for passengers or for fuel. Short tons would be replaced by the number of passengers or barrels of fuel being shipped. Asset and port capacities would also be changed to reflect the different units. The model could then be run three times, once for each major type of movement requirement.

e. If the length of a time period were made a parameter, then longer problems could be solved. The modeler must also recognize and weigh the effects this technique will have on his solution's resolution.

f. Develop a means by which certain movement requests could be "flagged" for movement by a specific transportation mode. Before the problem even starts, we know that certain commodities such as tanks must be moved by sealift. If we can represent this fact to the optimizer, then not only have we reduced the number of decision variables, we have also more realistically modeled the problem.

6.   The current version of the ARA is written in FORTRAN 77. It may be beneficial to convert the code to some other language, such as PASCAL. The ability of PASCAL to dynamically manage memory and to structure "mixed mode" (character or numeric) arrays would be two improvements enabling the ARA to reduce larger problems.

7.   The ARA is reasonably efficient because of the way it stores or distributes information out among a great many "short stacks." This storage and data retrieval technique is referred to as "bucketing." An improvement over this technique would be to create an appropriate "hashing" function [Ref. 21]. The hashing function would improve the efficiency of the ARA two ways. It would decrease run time because we would no longer be searching through stacks looking for information. While the hashing function does not guarantee direct accessing of your data, it can approach it. The hashing function would also assist in reducing the amount of storage required and thus free the programmer from the requirement to dimension off huge blocks of memory to accommodate the required number of stacks.

D. STUDY CONCLUSIONS

1. Linear programming with variable reduction is a viable alternative for modeling and solving both small- and medium-sized deployment plans. Since a large portion of plans fall into this size category, consideration should be given to continual development of this approach as an alternative solution technique. An LP may never approach being able to solve the largest of deployment problems. There are, however, advantages to being able to solve the smaller- and medium-sized problems as an LP. These advantages include: (a) the availability of commercial LP solvers which will make the system more portable and much less expensive to develop than specialized decomposition algorithms, (b) deployment plan evaluations and modifications will be much easier to resolve since the LP dual variables are readily interpretable, and (c) model enhancements and modifications will be much simpler to implement and test.

2. The ARA is clearly the most significant product of this study. Its ability to reduce the size of large problems without affecting the optimal solution was the key factor leading to the success of this thesis. Regardless of the formulation and solution technique eventually used by MODES (decomposition or LP), the benefits of reducing the size of the problem in this manner can be realized. The greatest benefit will most likely be realized when some version of the ARA is used in conjunction with a properly functioning decomposition formulation. It may be possible to solve even the largest of deployment problems once these two methods are applied in tandem.

66

3. Even though the sealift flows have been "spiked," there are still too many "little ships" running around. While this representation of sealift is an improvement over the previous ships, which appeared as "pipes," this representation of sealift is still not adequate. The integer programming effort of Captain Mike Lally may provide the solution to the sealift portion of this model [Ref. 2].

4. The GAMS modeling language is an excellent developmental tool. The ability to proceed directly from the mathematical representation of a model to an optimal solution saves the analyst/modeler countless programming hours, and allows him to try out many alternate formulations

5. The organization and logic structure of the Arc Reduction Algorithm has the potential to be applied to a great many management and/or complicated decision-making problems. The ARA's ability to intelligently seek out a set of "good" paths is analogous to any problem where there is a "sequence" of decisions or alternatives to be considered. In each of these cases, a different set of acceptability rules could be developed that would enable the decision maker to reduce the domain of his decision set down to a more reasonable size.

LIST OF REFERENCES

1. Joint Deployment Agency. <u>JDS Mode Optimization and Deployment Estimation Subsystem (MODES) System Description</u>. 27 September 1984.

2. Laily, M., <u>Strategic Allocation of Sealift: A GAMS-Based integer Programming Approach</u>. M.S. Thesis. Naval Postgraduate School. Monterey, CA. September 1987.

3. Joint Deployment Agency. <u>MODES OT&E Plan (Draft)</u>, 12 December 1986

4. Jarvis, J. J., Ratliff, H.D., Eisenstein, D. E., Iyer, A. V., Nulty, W. G., and Trick, M. A., <u>System for Closure Optimization Planning and Evaluation (SCOPE)</u>, PDRC Report 84-09, Georgia Institute of Technology. 1985.

5. Jarvis, J. J., McCroan, K. L., Nulty, W. G., Ratliff, H. D., and Trick. M. A., <u>Modifications to Sealift Capability Estimation in MRMATE</u>, PDRC Report 86-06, Georgia Institute of Technology. 1986.

6. Staniec. C. J., <u>Design and Solution of an Ammunition Distribution Model By a Resource-Directive Multicommodity Network Flow Algorithm</u>, M.S. Thesis. Naval Postgraduate School. Monterey, CA. September 1984.

7. Assad, A. A., "Multicommodity Network Flows— A Survey." <u>Networks</u>, Vol. 8, pp. 37-91, 1978.

8. Kennington, J. L., "A Survey of Linear Cost Multicommodity Network Flows." <u>Operations Research</u>, Vol. 26, No. 2, pp. 209-236, March-April 1978.

9. Bradley, G. H., Brown, G. G., and Graves, G. W., "Design and Implementation of Large Scale Primal Transshipment Algorithms," <u>Management Science</u>, Vol. 24, No. 1, p. 1. September 1977.

10. Bazaraa, J. S., and Jarvis, J. J., <u>Linear Programming and Network Flows</u>, pp. 404-405, John Wiley & Sons, Inc., New York, 1977.

11. Geoffrion, A. M., and Graves, G. W., "Multicommodity Distribution System Design by Benders Decomposition," _Management Science_, Vol. 20, No. 5, pp. 822–823, January 1974.

12. Liebman, J., Lasdon, L., Schrage, L., and Warren, A., _Modeling and Optimization With GINO_, pp. 36–37, The Scientific Press, Palo Alto, CA, 1986.

13. Iyer, A. V., Jarvis, J. J., and Ratliff, H. D., _Network Aggregation Concepts_, PDRC Report 85–04, Georgia Institute of Technology, 1986.

14. Aho, A. V., Hopcroft, J. E., and Ullman, J. D., _Data Structures and Algorithms_, p. 215, Addison–Wesley Publishing Co., Reading, PA, 1983.

15. Ketron Management Science, Inc., _MPS III Mathematical Programming System–General Description_, January 1987.

16. Rosenthal, R. E., "Review of the GAMS/MINOS Modeling Language and Optimization Program," _OR/MS Today_, Vol. 13, No. 3, pp. 24–32, June 1986.

17. Schrage, L., _Linear, Integer, and Quadratic Programming with LINDO_, pp. 266–269, Scientific Press, Palo Alto, CA, 1984.

18. Kendrick, D., and Meeraus, A., _GAMS An Introduction_, pp. 8–15, World Bank, Washington, D.C., January 1987.

19. Ford, L. K., and Fulkerson, D. R., _Flows in Networks_, pp. 6–8, Rand, August 1962.

20. Brown, G. B., Graves, G. W., and Ronen, David, "Scheduling Ocean Transportation of Crude Oil," _Management Science Magazine_, Vol. 33, No. 3, p. 341, March 1987.

21. Williamson, S. G., _Combinatorics for Computer Science_, pp. 8–9, Computer Science Press, Rockville, Maryland, 1985.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Mr. Tony Brooke Analytic Support Unit Development Research Department The World Bank 1818 H. Street, N.W. Washington, DC 20433	1
4.	Professor Gerald G. Brown, Code 55Bw Department of Operations Research Naval Postgraduate School Monterey, California 93943-5004	1
5.	Professor Siriphong Lawphongpanich, Code 55 Department of Operations Research Naval Postgraduate School Monterey, California 93943-5004	1
6.	Mr. Alexander Meeraus Analytic Support Unit Development Research Department The World Bank 1818 H. Street, N.W. Washington, DC 20433	1
7.	Professor Richard E. Rosenthal, Code 55 Rl Department of Operations Research Naval Postgraduate School Monterey, California 93943-5004	5
8.	U.S. Transportation Command TCJD-S-TD MacDill AFB, Florida 33608	2

9.   Captain K. Steven Collier   10
        Commandant
        U.S. Army Armor School
        ATTN: ATSB-CD-SD
        Ft. Knox, Kentucky 40121-5215

END
DATE
FILMED
JAN
1988